

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR UNITED STATES PATENT

FOR

**MANAGEMENT INFORMATION BASE FOR A  
MULTI-DOMAIN NETWORK ADDRESS TRANSLATOR**

Inventor:

**Richard Crump**  
295 Beacon Street  
Boston, MA 02117

**Timothy Cunningham**  
566 Old Dunstable Road  
Groton, MA 01450

**Joseph Flaherty**  
69 West Main Street  
Westborough, MA 01581-2516

**Manish Patel**  
67 Sequoia Drive  
Tyngsboro, MA 01879

Attorney Docket No.: 2204/A39

Client Reference No.: BA0391

Attorneys:

**BROMBERG & SUNSTEIN LLP**  
125 Summer Street  
Boston, MA 02110  
(617) 443-9292

## MANAGEMENT INFORMATION BASE FOR A MULTI-DOMAIN NETWORK ADDRESS TRANSLATOR

### 5 CROSS-REFERENCE TO RELATED APPLICATIONS

This patent application may be related to the following commonly owned U.S. patent application, which is hereby incorporated by reference in its entirety:

6 Application No. 09/274,940 entitled NETWORK ADDRESS TRANSLATION IN A  
10 NETWORK HAVING MULTIPLE OVERLAPPING ADDRESS DOMAINS, filed on March  
23, 1999 in the names of Timothy Cunningham, Thomas Meehan, and Manish Patel  
15 (Attorney Docket No. 2204/116).

### FIELD OF THE INVENTION

15 The present invention relates generally to communication networks, and, more particularly, to managing a multi-domain network address translator for translating network addresses in a network having multiple overlapping address domains.

20

### BACKGROUND OF THE INVENTION

25 In today's information age, communication networks are increasingly used for transferring information among a multitude of communication devices. As demand for communication services continues to grow, the demand on these communication networks for carrying increasing amounts of information at increasing speeds continues to grow. Therefore, communication networks are evolving to more efficiently handle these increased demands.

30

In a common networking model, a large communication network is typically constructed by segregating the multitude of communication devices into a number of subnetworks, and internetworking the subnetworks over a high-speed backbone network. In such a communication network, each communication device is typically assigned a network address that is used for routing packets between a source communication device and a

destination communication device within the communication network. In order to permit efficient use of these network addresses, the communication network may be logically divided into multiple address domains. Network addresses are required to be unique within a particular address domain, but are not required to be unique across multiple address domains.

5        Unfortunately, when the communication network is logically divided into multiple address domains having overlapping network addresses, a particular network address may map to multiple communication devices, in which case the network address does not uniquely identify one communication device within the communication network. Such an overlapping network address cannot be used as the destination address of a packet because it is ambiguous as to the destination communication device for the packet. Likewise, such an overlapping network address, when used as the source address of a packet, is ambiguous as to the source communication device for the packet.

Thus, a need has remained for a network address translation technique for resolving ambiguous network addresses across multiple overlapping address domains.

## **SUMMARY OF THE INVENTION**

In accordance with one aspect of the invention, a management information base (MIB) for a multi-domain network address translator provides management objects for configuring and controlling the multi-domain network address translator. The MIB includes management objects for defining a domain-specific source address filter range, which is a range of addresses used to detect domain-specific packets that require domain-specific network address translation forwarding. The domain-specific source address filter management objects include a beginning address, a prefix length, a domain indicator, and a domain-specific translation pool indicator. The MIB also includes management objects for defining a domain-specific translation pool, which is a range of addresses from which domain-specific translation addresses are selected for domain-specific network address translation forwarding. The domain-specific translation pool management objects include a beginning address, a prefix length, and a domain indicator.

## BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing and other objects and advantages of the invention will be appreciated  
5 more fully from the following further description thereof with reference to the accompanying  
drawings wherein:

FIG. 1 is a block diagram showing an exemplary communication network having  
multiple overlapping address domains in accordance with an embodiment of the present  
invention;

FIG. 2A is a diagram showing an exemplary source address translation table for a first  
source (inbound) address domain in the communication network in accordance with an  
embodiment of the present invention;

FIG. 2B is a diagram showing an exemplary source address translation table for a  
second source (inbound) address domain in the communication network in accordance with  
an embodiment of the present invention;

FIG. 2C is a diagram showing an exemplary source address translation table for a  
third source (inbound) address domain in the communication network in accordance with an  
embodiment of the present invention;

FIG. 2D is a diagram showing an exemplary destination address translation table in  
20 accordance with an embodiment of the present invention;

FIG. 3 is a logic flow diagram showing exemplary network address translator logic for  
creating a source address translation table entry and a corresponding destination address  
translation table entry in accordance with an embodiment of the present invention;

FIG. 4 is a message flow diagram showing an exemplary message flow for resolving a  
25 domain name into a destination host global address in accordance with an embodiment of the  
present invention;

FIG. 5 is a logic flow diagram showing exemplary domain name system proxy logic  
for resolving a domain name into a destination host global address in accordance with an  
embodiment of the present invention;

FIG. 6 is a logic flow diagram showing exemplary network address translator logic for  
30 translating a destination host local address into a destination host global address as part of a

domain name resolution procedure in accordance with an embodiment of the present invention;

FIG. 7 is a message flow diagram showing an exemplary message flow for a first exemplary embodiment of the present invention;

5 FIG. 8 is a message flow diagram showing an exemplary message flow for a second exemplary embodiment of the present invention;

FIG. 9 is a logic flow diagram showing exemplary network address translator logic for performing network address translation on a packet in accordance with an embodiment of the present invention;

10 FIG. 10A is a logic flow diagram showing exemplary destination address translation logic for translating a destination address from a destination host global address into a destination host local address in a destination (outbound) address domain in accordance with an embodiment of the present invention;

15 FIG. 10B is a logic flow diagram showing exemplary source address translation logic for translating a source address from a source host local address into a source host global address for the destination (outbound) address domain in accordance with an embodiment of the present invention;

20 FIG. 11A is a block diagram showing the relevant logic blocks of an exemplary network address translator in accordance with an embodiment of the present invention;

FIG. 11B is a block diagram showing the relevant logic blocks of an exemplary packet processor in accordance with an embodiment of the present invention; and

25 FIGs. 12A-12P show an exemplary management information base for managing a multi-domain network address translator in accordance with an embodiment of the present invention.

## DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT

30 As discussed above, a need has remained for a network address translation technique for resolving ambiguous network addresses across multiple overlapping address domains. In accordance with the present invention, a network address translator (NAT) maps an

overlapping domain-specific network address in a first address domain (referred to hereinafter as a "local address") to a unique global address that is specific to a second address domain. Thus, the overlapping network address in the first address domain may map to multiple global addresses, where each global address is unique to one of the other address domains.

5 The NAT uses the network address mappings to translate the source address and/or the destination address of a packet before the packet is routed from the source communication device (referred to hereinafter as the "source host") to its intended destination communication device (referred to hereinafter as the "destination host"). Specifically, the NAT translates the destination address from a destination host global address (which uniquely identifies both the source address domain and the destination address domain) to its corresponding destination host local address, upon determining that the destination address requires an address translation. Likewise, the NAT translates the source address from an overlapping source host local address to a unique source host global address based upon the source address domain and the destination address domain, upon determining that the source address requires an address translation. By translating the source address and/or the destination address, the resulting packet is able to be routed to the destination host in the destination address domain using the destination host local address, and the destination host is able to uniquely identify the source host for the packet using the unique source host global address.

20 Network address translation has been used in the past to allow local addresses to be reused within a communication network. One prior art network address translation technique is described in an Internet Engineering Task Force (IETF) document entitled The IP Network Address Translator (NAT), by K. Egevang and P. Francis (May 1994). In a typical prior art embodiment, the NAT maps a local address to a single global address irrespective of the destination address domain. Thus, when the local address is included as the source address in 25 a packet, the NAT translates the local address into the global address without regard for the destination address domain before forwarding the packet to the destination host. Likewise, when the global address is included as the destination address in a packet, the NAT translates the global address into the local address before routing the packet to the destination host.

30 In certain networking models, it is desirable for the local address to map to a different global address for each destination address domain. The present invention provides a

network address translation technique that allows the local address to be mapped to a different global address for each destination address domain. Specifically, a preferred NAT maps the local address to a different global address for each destination address domain, where each global address is unique within the communication network and maps uniquely to the local address. When the local address is included as the source address in a packet transmitted to a particular destination address domain, the preferred NAT translates the local address into the specific global address for the destination address domain. Likewise, when a global address is included as the destination address of a packet, the preferred NAT translates the global address into the local address.

In a preferred embodiment of the present invention, the NAT performs address translations for routing packets in a communication network having multiple overlapping address domains, such as the exemplary communication network 100 as shown in FIG. 1. In the exemplary communication network 100, there are three (3) hosts that share a common network address A across three (3) overlapping address domains, namely host X 110 in address domain 1, host Y 120 in address domain 2, and host Z 130 in address domain 3. There is also one (1) host with a non-overlapping network address, namely host B 140 in address domain 4. Thus, the address A represents the local address for each host that uniquely identifies a particular host within its own address domain. Unfortunately, the address A is ambiguous within the entire communication network 100, since it does not uniquely identify a specific host within the entire communication network 100. Therefore, the communication network 100 includes a NAT 102 to perform, among other things, the network address translations needed to resolve the ambiguity of the address A within the communication network 100.

In order for a host in an address domain q to reference a host in an address domain p having the overlapping address A, the NAT 102 maps the overlapping address A from the address domain p to a global address that is unique to the address domain q and is also unique within the communication network 100. For convenience, the global address for a host having the local address A in the address domain p when referenced from a host in the address domain q is represented by the symbol Apq. Thus, Apq is the global address for the address A in address domain p when referenced from address domain q.

Thus, with reference to the example shown in FIG. 1, the NAT 102 typically maintains at least the following global address mappings:

A12 is the host X global address when referenced from address domain 2;  
5 A13 is the host X global address when referenced from address domain 3;  
A14 is the host X global address when referenced from address domain 4;  
A21 is the host Y global address when referenced from address domain 1;  
A23 is the host Y global address when referenced from address domain 3;  
A24 is the host Y global address when referenced from address domain 4;  
A31 is the host Z global address when referenced from address domain 1;  
A32 is the host Z global address when referenced from address domain 2; and  
A34 is the host Z global address when referenced from address domain 4.

In a preferred embodiment of the present invention, the NAT 102 maintains a separate source address translation table for each overlapping address domain, and maintains a single destination address translation table. Each source address translation table maps the overlapping network addresses in the source address domain to the corresponding global addresses for each of the potential destination address domains. The destination address translation table maps the global addresses to their corresponding local addresses.

20 Thus, in the example shown in FIG. 1, the preferred NAT 102 maintains three (3) source address translation tables and one (1) destination address translation table. An exemplary source address translation table for address domain 1, shown in FIG. 2A, maps the host X local address (i.e., A) to the respective host X global addresses for domains 2, 3, and 4 (i.e., A12, A13, and A14, respectively). An exemplary source address translation table for address domain 2, shown in FIG. 2B, maps the host Y local address (i.e., A) to the respective host Y global addresses for domains 1, 3, and 4 (i.e., A21, A23, and A24, respectively). An exemplary source address translation table for address domain 3, shown in FIG. 2C, maps the host Z local address (i.e., A) to the respective host Z global addresses for domains 1, 2, and 4 (i.e., A31, A32, and A34, respectively). No source address translation table is maintained for address domain 4, since, in this example, address domain 4 has no overlapping network

addresses. An exemplary destination address translation table, shown in FIG. 2D, maps the global addresses to their corresponding local addresses.

It should be noted that, by maintaining the source (inbound) domain and the destination (outbound) domain in the tables, it is possible to combine all source address translation tables into a single source address translation table. It should also be noted that, by maintaining the source (inbound) domain and the destination (outbound) domain in the tables, the source table(s) and the destination table provide redundant information, such that, for example, the source address translation table(s) can be searched in reverse to obtain a local address corresponding to a particular global address, or the destination address translation table can be searched in reverse to obtain a global address corresponding to a particular local address in the source address domain. These and other alternative embodiments will be apparent to a skilled artisan.

In order to transfer a packet from the source host in the source address domain to the destination host in the destination address domain using network address translation, the appropriate entries must be created in the source address table(s) and the destination address table. Specifically, for any network address that must be translated, there must be a source address translation table entry mapping the source host local address in the source address domain to a unique source host global address for the destination address domain, and there must be a destination address translation table entry mapping the source host global address for the destination address domain back to the source host local address in the source address domain.

Typically, the address translation table entries are created dynamically by the NAT 102, although the address translation table entries may alternatively be created manually. In order for the NAT 102 to create a source address translation table entry and its corresponding destination address translation table entry, the NAT 102 is provided with at least a source host local address, a source address domain identifier, and a destination address domain identifier. The NAT 102 selects a source host global address from a pool of global network addresses, and creates the address translation table entries. Specifically, the NAT 102 creates a source address translation table entry mapping the source host local address in the source address domain to the selected source host global address for the destination address domain, and

creates a destination address translation entry mapping the selected source host global address for the destination address domain to the source host local address in the source address domain.

5 FIG. 3 is a logic flow diagram showing exemplary NAT 102 logic for creating a source address translation table entry and its corresponding destination address translation table entry. Beginning at step 302, the logic receives a host local address, a first address domain identifier identifying the host address domain, and a second address domain identifier identifying an address domain from which the host is referenced, in step 304. The logic proceeds to select a unique global address for the host, in step 306, preferably from a pool of global addresses maintained by the NAT 102. Upon selecting the unique global address in step 306, the logic creates a source address translation table entry in the first address domain's source address translation table, in step 308, and a corresponding destination address translation table entry, in step 310. The source address translation table entry maps the host local address in the first address domain to the selected global address for the second address domain. The destination address translation table entry maps the selected global address to the host local address in the first address domain. The logic terminates in step 399.

20 In a preferred embodiment of the present invention, the NAT 102 dynamically creates certain address translation table entries as part of a domain name resolution procedure, and dynamically creates other address translation entries as part of a packet processing procedure (described in more detail below). The domain name resolution procedure is described in the related U.S. patent application entitled DOMAIN NAME RESOLUTION IN A NETWORK HAVING MULTIPLE OVERLAPPING ADDRESS DOMAINS, which was incorporated by reference above. The domain name resolution procedure enables the source host to obtain a destination host global address for the destination host based upon a domain name of the destination host. More particularly, in order for the source host to transmit a packet to the destination host, the source host is provided with a domain name that is associated with the destination host. The domain name uniquely identifies the destination host, although the domain name is not a network address. The source host invokes the domain name resolution procedure in order to resolve the domain name into the destination host global address.

25 Because the destination host local address may overlap with other addresses when the

communication network includes multiple overlapping address domains, a preferred domain name resolution procedure utilizes network address translation to translate the overlapping destination host local address into a unique destination host global address.

Specifically, in order for the source host to resolve the destination host domain name into the destination host global address, the source host sends a domain name resolution request to a local DNS Server in the source address domain. The domain name resolution request includes, among other things, a source address equal to the source host local address and the domain name associated with the destination host. The local DNS Server in the source address domain maintains a cache of domain name/network address mappings for hosts within the source address domain. Upon receiving the domain name resolution request from the source host, the local DNS Server in the source address domain determines the destination host domain name corresponds to a host in a different address domain. The local DNS Server therefore sends a domain name resolution request to the DNS Proxy 104.

The DNS Proxy 104 performs domain name resolution across multiple address domains. Upon receiving the domain name resolution request from the local DNS Server in the source address domain, the DNS Proxy 104 determines the destination address domain for the destination host domain name, and sends a domain name resolution request to the local DNS Server in the destination address domain. The local DNS Server in the destination address domain maintains a cache of domain name/network address mappings for hosts within the destination address domain. Upon receiving the domain name resolution request from the DNS Proxy 104, the local DNS Server in the destination address domain resolves the domain name, and returns the destination host local address to the DNS Proxy 104.

Upon receiving the destination host local address from the local DNS Server in the destination address domain, the DNS Proxy 104 sends a translation request to the NAT 102 to translate the destination host local address into a unique destination host global address. The translation request includes, among other things, a source address domain identifier, the destination host local address, and a destination address domain identifier.

The NAT 102 maintains a pool of global network addresses, and also maintains a number of address translation entries, where each address translation entry maps a local host address from one address domain to a global address that is specific to another address

domain. Upon receiving the translation request from the DNS Proxy 104, the NAT 102 first determines whether there is an existing address translation table entry mapping the destination host local address to a destination host global address that is specific to the source address domain. If there is not an existing address translation table entry mapping the destination host local address to a destination host global address that is specific to the source address domain, then the NAT 102 creates the appropriate address translation table entries. Specifically, the NAT 102 selects a destination host global address from the pool of global network addresses, and creates both a source address translation entry and a corresponding destination address translation entry mapping the destination host local address to the destination host global address specifically for the source address domain. The source address translation table entry includes a Source Local Address field equal to the destination host local address, a Source Address Domain field equal to the destination address domain, a Destination Address Domain field equal to the source address domain, and a Source Global Address field equal to the selected destination host global address. The corresponding destination address translation table entry includes a Destination Global Address field equal to the selected destination host global address, a Source Address Domain field equal to the source address domain, a Destination Address Domain field equal to the destination address domain, and a Destination Local Address field equal to the destination host local address. The NAT 102 sends a translation response to the DNS Proxy 104 including the destination host global address.

Upon receiving the translation response from the NAT 102, the DNS Proxy 104 sends a domain name resolution response to the local DNS Server in the source address domain including the destination host global address. The local DNS Server in the source address domain, in turn, sends a domain name resolution response to the source host including the destination host global address. Thus, the domain name associated with the destination host is resolved into a unique destination host global address that the source host can use to transmit a packet to the destination host.

FIG. 4 is a message flow diagram showing an exemplary message exchange among the source host in the source address domain, the local DNS Server in the source address domain, the DNS Proxy 104, the local DNS Server in the destination address domain, and the

NAT 102 for resolving the destination host domain name into the unique destination host global address. Specifically, the source host sends a domain name resolution request message 402 to the local DNS Server in the source address domain including, among other things, a source address equal to the source host local address and the destination host domain name. 5 Since the local DNS Server in the source address domain is unable to resolve the destination host domain name, the local DNS Server in the source address domain sends a domain name resolution request message 403 to the DNS Proxy 104. The DNS Proxy 104, in turn, sends a domain name resolution request message 404 to the local DNS Server in the destination address domain including, among other things, the destination host domain name. Upon receiving the domain name resolution request message 404, the local DNS Server in the destination address domain resolves the destination host domain name into its corresponding destination host local address, and sends a domain name resolution response message 406 to the DNS Proxy 104 including, among other things, the destination host local address. Upon receiving the domain name resolution response message 406 including the destination host local address, the DNS Proxy 104 sends a translation request message 408 to the NAT 102 including, among other things, the source address domain identifier, the destination host local address, and the destination address domain identifier. Upon receiving the translation request message 408, the NAT 102 creates the appropriate address translation entries, if necessary, and sends a translation response message 410 to the DNS Proxy 104 including, among other things, the destination host global address. The DNS Proxy 104 sends a domain name resolution response message 412 to the local DNS Server in the source address domain, which, in turn, sends a domain name resolution response message 414 to the source host including, among other things, the destination host global address. 20

FIG. 5 is a logic flow diagram showing exemplary DNS Proxy 104 logic for resolving a domain name in a network having multiple overlapping address domains. Beginning in step 25 502, the DNS Proxy 104 receives the domain name resolution request message 403, in step 504. The domain name resolution request message 403 includes, among other things, the destination host domain name associated with the destination host in the destination address domain.

30 The DNS Proxy 104 sends the domain name resolution request message 404 to the

local DNS Server in the destination address domain, in step 506. The domain name resolution request message 404 includes, among other things, the destination host domain name. The DNS Proxy 104 then monitors for the domain name resolution response message 406 from the local DNS Server in the destination address domain including the destination host local address.

Upon receiving the domain name resolution response message 406 including the destination host local address, in step 508, the DNS Proxy 104 sends the translation request message 408 to the NAT 102, in step 510. The translation request message 408 includes, among other things, the source address domain identifier, the destination host local address, and the destination address domain identifier. The DNS Proxy 104 then monitors for the translation response message 410 from the NAT 102 including the destination host global address.

Upon receiving the translation response message 410 from the NAT 102, in step 512, the DNS Proxy 104 sends the domain name resolution response message 412, in step 514. The domain name resolution response message 412 includes, among other things, the destination host global address. The DNS Proxy 104 logic terminates in step 599.

FIG. 6 is a logic flow diagram showing exemplary NAT 102 logic for translating the destination host local address into the unique destination host global address that is specific to the source address domain as part of the domain name resolution procedure. Beginning in step 602, the NAT 102 receives the translation request message 408 from the DNS Proxy 104, in step 604. The translation request message 408 includes, among other things, the source address domain identifier, the destination host local address, and the destination address domain identifier. The NAT 102 then searches the address translation entries for an address translation entry mapping the destination host local address in the destination address domain to a unique destination host global address for the source address domain, in step 606. If the NAT 102 finds such an address translation entry (YES in step 608), then the NAT 102 proceeds to step 618. Otherwise (NO in step 608), the NAT 102 creates the source address translation table entry and the corresponding destination address translation entry.

In order to create the address translation table entries, the NAT 102 first selects a unique destination host global address, in step 612, preferably from a pool of global network

addresses maintained by the NAT 102. Upon selecting the destination host global address in step 612, the NAT 102 creates a source address translation table entry in the destination address domain's source address translation table, in step 614, and a corresponding destination address translation table entry, in step 616. The source address translation table entry maps the destination host local address in the destination address domain to the destination host global address for the source address domain. The destination address translation table entry maps the destination host global address to the destination host local address in the destination address domain.

In step 618, the NAT 102 sends the translation response message 412 including the destination host global address. The NAT 102 logic terminates in step 699.

Once the source host has obtained the destination host global address, either through domain name resolution or some other means, the source host transmits a packet including, as the destination address, the destination host global address for the source address domain, and, as the source address, the source host local address. The destination address uniquely identifies the destination host within the communication network 100. However, the source address is an ambiguous address within the communication network 100.

Upon receiving the packet, the NAT 102 uses the destination address to determine, among other things, the destination address domain for the packet. However, the NAT 102 cannot simply route the packet to the destination host over the destination address domain using traditional routing techniques. This is because the destination address in the packet is not equal to the destination host local address in the destination address domain, and, consequently, the packet would not be received by the destination host in the destination address domain.

Therefore, after determining that the packet requires address translation, the NAT 102 translates the destination address from the destination host global address into the destination host local address. In order to translate the destination address, the NAT 102 uses the destination address translation table to obtain the destination host local address, specifically by finding the destination address translation table entry corresponding to the destination host global address and obtaining therefrom the destination host local address.

In certain situations, the NAT 102 may also have to translate the source address in the

packet from the source host local address in the source address domain into a unique source host global address for the destination address domain. Such an address translation is required when the source host local address is an overlapping address within the communication network. The source address translation is done so that the destination host receives a globally unique source address that uniquely identifies the source host within the communication network. The source address can therefore be used by the destination host, for example, to send a response packet to the source host.

In order to translate the source address, the NAT 102 first determines both the source domain (either implicitly based upon the interface over which the packet is received or explicitly from the destination address translation table entry) and the destination domain (from the destination address translation table entry) for the packet. The NAT 102 then searches the address translation entries to find an address translation entry mapping the source host local address in the source address domain to a source host global address for the destination address domain. If the NAT 102 finds such an address translation entry, then the NAT 102 translates the source address in the packet by extracting the source host global address from the address translation entry and replacing the source host local address in the packet with the source host global address. However, if there is no address translation entry mapping the source host local address in the source address domain to a source host global address for the destination address domain, then the NAT 102 dynamically allocates a source host global address for the destination address domain, creates the appropriate address translation entries, and translates the source address in the packet by replacing the source host local address in the packet with the dynamically allocated source host global address.

More specifically, the NAT 102 first selects the source host global address from a pool of network addresses. The NAT 102 then creates a source address translation table entry in the source address translation table for the source address domain and a corresponding destination address translation table entry in the destination address translation table. The source address translation table entry includes a Source Local Address field equal to the source host local address, a Source Address Domain field equal to the source address domain, a Destination Address Domain field equal to the destination address domain, and a Source Global Address field equal to the selected source host global address. The corresponding

destination address translation table entry includes a Destination Global Address field equal to the selected source host global address, a Source Address Domain field equal to the destination address domain, a Destination Address Domain field equal to the source address domain, and a Destination Local Address field equal to the source host local address.

5 After translating either the destination address, the source address, or both addresses in the packet, the NAT forwards the translated packet to the destination host over the destination address domain.

The network address translations described above can be demonstrated by example. Two examples are set forth below. The first example follows the network address translations of a packet sent by the host X 110 to the host Y 120 and a corresponding response packet sent by the host Y 120 back to the host X 110. The second example follows the network address translations of a packet sent by the host X 110 to the host B 140 and a corresponding response packet sent by the host B 140 back to the host X 110. In these examples, it is assumed that the host X 110 has obtained the destination address using domain name resolution or some other means. For convenience, the convention (S, D) is used to indicate a packet having source address S and destination address D.

FIG. 7 is a message flow diagram showing an exemplary packet exchange between the host X 110 in the address domain 1 and the host Y 120 in the address domain 2. The host X 110 transmits the packet 702 including, as the source address, the host X local address (i.e., 20 A), and, as the destination address, the host Y global address for address domain 1 (i.e., A21). The host Y global address A21 uniquely identifies the host Y 120 within the communication network 100. However, the host X local address A is ambiguous within the communication network 100, since it does not uniquely identify the host X 110.

Upon receiving the packet 702, the NAT 102 determines that both the source address 25 and the destination address require address translation. In order to translate the destination address, the NAT 102 uses the destination address translation table shown in FIG. 2D to find the destination address translation table entry 226 corresponding to the destination address A21, and obtains therefrom the host Y local address A. In order to translate the source address, the NAT 102 obtains the destination address domain from the destination address 30 translation table entry 226 (i.e., address domain 2), and also determines the source address

domain (i.e., address domain 1) either implicitly based upon the interface over which the packet 702 is received or explicitly from the destination address translation table entry 226. The source address domain indicates the particular source address translation table required for the source address translation, which, in this example, is the source address translation table for address domain 1 shown in FIG. 2A. The NAT 102 finds the source address translation table entry 202 corresponding to the host X local address for destination (outbound) address domain 2, and obtains therefrom the host X global address for address domain 2 (i.e., A12). The NAT 102 then formats the packet 704 including, as the source address, the host X global address for address domain 2 (i.e., A12), and, as the destination address, the host Y local address (i.e., A). The NAT 102 forwards the packet 704 to the host Y 120 over the address domain 2.

Upon receiving the packet 704, the host Y 120 may transmit a response packet 706 including, as the source address, the host Y local address (i.e., A), and, as the destination address, the host X global address for address domain 2 (i.e., A12), typically copied from the source address of the packet 704. The host X global address A12 uniquely identifies the host X 110 within the communication network 100. However, the host Y local address A is ambiguous within the communication network 100, since it does not uniquely identify the host Y 120.

Upon receiving the packet 706, the NAT 102 determines that both the source address and the destination address require address translation. In order to translate the destination address, the NAT 102 uses the destination address translation table shown in FIG. 2D to find the destination address translation table entry 220 corresponding to the destination address A12, and obtains therefrom the host X local address A. In order to translate the source address, the NAT 102 obtains the destination address domain from the destination address translation table entry 220 (i.e., address domain 1), and also determines the source address domain (i.e., address domain 2) either implicitly based upon the interface over which the packet 706 is received or explicitly from the destination address translation table entry 220. The source address domain indicates the particular source address translation table required for the source address translation, which, in this example, is the source address translation table for address domain 2 shown in FIG. 2B. The NAT 102 finds the source address

translation table entry 208 corresponding to the host Y local address for destination (outbound) address domain 1, and obtains therefrom the host Y global address for address domain 1 (i.e., A21). The NAT 102 then formats the packet 708 including, as the source address, the host Y global address for address domain 1 (i.e., A21), and, as the destination address, the host X local address (i.e., A). The NAT 102 forwards the packet 708 to the host X 110 over the address domain 1.

FIG. 8 is a message flow diagram showing an exemplary packet exchange between the host X 110 in the address domain 1 and the host B 140 in the address domain 4. The host X 110 transmits the packet 802 including, as the source address, the host X local address (i.e., A), and, as the destination address, the host B network address (i.e., B). The host B network address B uniquely identifies the host B 140 within the communication network 100. However, the host X local address A is ambiguous within the communication network 100, since it does not uniquely identify the host X 110.

Upon receiving the packet 802, the NAT 102 determines that only the source address requires address translation. In order to translate the source address, the NAT 102 determines the destination address domain, for example, by finding the destination address translation table entry 238 in the destination address translation table, and obtaining therefrom the destination (outbound) domain (i.e., address domain 4). The NAT 102 also determines the source address domain (i.e., address domain 1) implicitly based upon the interface over which the packet 502 is received (there is no explicit source address domain associated with the network address B). The source address domain indicates the particular source address translation table required for the source address translation, which, in this example, is the source address translation table for address domain 1 shown in FIG. 2A. The NAT 102 finds the source address translation table entry 206 corresponding to the host X local address for destination (outbound) address domain 4, and obtains therefrom the host X global address for address domain 4 (i.e., A14). The NAT 102 then formats the packet 804 including, as the source address, the host X global address for address domain 4 (i.e., A14), and, as the destination address, the host B network address (i.e., B). The NAT 102 forwards the packet 804 to the host B 140 over the address domain 4.

Upon receiving the packet 804, the host B 140 may transmit a response packet 806

including, as the source address, the host B network address (i.e., B), and, as the destination address, the host X global address for address domain 4 (i.e., A14), typically copied from the source address of the packet 804. The host X global address A14 uniquely identifies the host X 110 within the communication network 100. The host B network address B is unambiguous within the communication network 100.

Upon receiving the packet 806, the NAT 102 determines that only the destination address requires address translation. In order to translate the destination address, the NAT 102 uses the destination address translation table shown in FIG. 2D to find the destination address translation table entry 224 corresponding to the destination address A14, and obtains therefrom the host X local address A. The NAT 102 then formats the packet 808 including, as the source address, the host B network address B, and, as the destination address, the host X local address A. The NAT 102 forwards the packet 808 to the host X 110 over the address domain 1.

FIG. 9 is a logic flow diagram showing exemplary NAT 102 logic for processing a packet received from the source host. Beginning in step 902, the NAT 102 receives from the source host a packet including a source address equal to a source host local address and a destination address equal to a destination host global address, in step 904. The destination host global address is, by definition, a unique address within the communication network 100, although the destination host global address may or may not need to be translated into a destination host local address in the destination address domain. The source host local address may be either a unique address within the communication network 100 or an overlapping address that needs to be translated into a source host global address for the destination address domain.

Therefore, upon receiving the packet in step 904, the NAT 102 determines whether the destination address requires translation, in step 906. If the destination address requires translation (YES in step 908), then the NAT 102 translates the destination address from the unique destination host global address to the destination host local address in the destination address domain, in step 910, as described in detail with respect to FIG. 10A below.

Whether or not the destination address requires translation, the NAT 102 also determines whether the source address requires translation, in step 912. If the source address

requires translation (YES in step 914), then the NAT 102 translates the source address from the overlapping source host local address to the unique source host global address for the destination address domain, in step 916, as described in detail with respect to FIG. 10B below.

5 After performing any required address translations, the NAT 102 forwards the translated packet into the destination address domain, in step 918. The NAT 102 logic terminates in step 999.

0  
5  
10  
15  
20  
25  
30  
35  
40  
45  
50  
55  
60  
65  
70  
75  
80  
85  
90  
95  
100  
105  
110  
115  
120  
125  
130  
135  
140  
145  
150  
155  
160  
165  
170  
175  
180  
185  
190  
195  
200  
205  
210  
215  
220  
225  
230  
235  
240  
245  
250  
255  
260  
265  
270  
275  
280  
285  
290  
295  
300  
305  
310  
315  
320  
325  
330  
335  
340  
345  
350  
355  
360  
365  
370  
375  
380  
385  
390  
395  
400  
405  
410  
415  
420  
425  
430  
435  
440  
445  
450  
455  
460  
465  
470  
475  
480  
485  
490  
495  
500  
505  
510  
515  
520  
525  
530  
535  
540  
545  
550  
555  
560  
565  
570  
575  
580  
585  
590  
595  
600  
605  
610  
615  
620  
625  
630  
635  
640  
645  
650  
655  
660  
665  
670  
675  
680  
685  
690  
695  
700  
705  
710  
715  
720  
725  
730  
735  
740  
745  
750  
755  
760  
765  
770  
775  
780  
785  
790  
795  
800  
805  
810  
815  
820  
825  
830  
835  
840  
845  
850  
855  
860  
865  
870  
875  
880  
885  
890  
895  
900  
905  
910  
915  
920  
925  
930  
935  
940  
945  
950  
955  
960  
965  
970  
975  
980  
985  
990  
995  
1000  
1005  
1010  
1015  
1020  
1025  
1030  
1035  
1040  
1045  
1050  
1055  
1060  
1065  
1070  
1075  
1080  
1085  
1090  
1095  
1100  
1105  
1110  
1115  
1120  
1125  
1130  
1135  
1140  
1145  
1150  
1155  
1160  
1165  
1170  
1175  
1180  
1185  
1190  
1195  
1200  
1205  
1210  
1215  
1220  
1225  
1230  
1235  
1240  
1245  
1250  
1255  
1260  
1265  
1270  
1275  
1280  
1285  
1290  
1295  
1300  
1305  
1310  
1315  
1320  
1325  
1330  
1335  
1340  
1345  
1350  
1355  
1360  
1365  
1370  
1375  
1380  
1385  
1390  
1395  
1400  
1405  
1410  
1415  
1420  
1425  
1430  
1435  
1440  
1445  
1450  
1455  
1460  
1465  
1470  
1475  
1480  
1485  
1490  
1495  
1500  
1505  
1510  
1515  
1520  
1525  
1530  
1535  
1540  
1545  
1550  
1555  
1560  
1565  
1570  
1575  
1580  
1585  
1590  
1595  
1600  
1605  
1610  
1615  
1620  
1625  
1630  
1635  
1640  
1645  
1650  
1655  
1660  
1665  
1670  
1675  
1680  
1685  
1690  
1695  
1700  
1705  
1710  
1715  
1720  
1725  
1730  
1735  
1740  
1745  
1750  
1755  
1760  
1765  
1770  
1775  
1780  
1785  
1790  
1795  
1800  
1805  
1810  
1815  
1820  
1825  
1830  
1835  
1840  
1845  
1850  
1855  
1860  
1865  
1870  
1875  
1880  
1885  
1890  
1895  
1900  
1905  
1910  
1915  
1920  
1925  
1930  
1935  
1940  
1945  
1950  
1955  
1960  
1965  
1970  
1975  
1980  
1985  
1990  
1995  
2000  
2005  
2010  
2015  
2020  
2025  
2030  
2035  
2040  
2045  
2050  
2055  
2060  
2065  
2070  
2075  
2080  
2085  
2090  
2095  
2100  
2105  
2110  
2115  
2120  
2125  
2130  
2135  
2140  
2145  
2150  
2155  
2160  
2165  
2170  
2175  
2180  
2185  
2190  
2195  
2200  
2205  
2210  
2215  
2220  
2225  
2230  
2235  
2240  
2245  
2250  
2255  
2260  
2265  
2270  
2275  
2280  
2285  
2290  
2295  
2300  
2305  
2310  
2315  
2320  
2325  
2330  
2335  
2340  
2345  
2350  
2355  
2360  
2365  
2370  
2375  
2380  
2385  
2390  
2395  
2400  
2405  
2410  
2415  
2420  
2425  
2430  
2435  
2440  
2445  
2450  
2455  
2460  
2465  
2470  
2475  
2480  
2485  
2490  
2495  
2500  
2505  
2510  
2515  
2520  
2525  
2530  
2535  
2540  
2545  
2550  
2555  
2560  
2565  
2570  
2575  
2580  
2585  
2590  
2595  
2600  
2605  
2610  
2615  
2620  
2625  
2630  
2635  
2640  
2645  
2650  
2655  
2660  
2665  
2670  
2675  
2680  
2685  
2690  
2695  
2700  
2705  
2710  
2715  
2720  
2725  
2730  
2735  
2740  
2745  
2750  
2755  
2760  
2765  
2770  
2775  
2780  
2785  
2790  
2795  
2800  
2805  
2810  
2815  
2820  
2825  
2830  
2835  
2840  
2845  
2850  
2855  
2860  
2865  
2870  
2875  
2880  
2885  
2890  
2895  
2900  
2905  
2910  
2915  
2920  
2925  
2930  
2935  
2940  
2945  
2950  
2955  
2960  
2965  
2970  
2975  
2980  
2985  
2990  
2995  
3000  
3005  
3010  
3015  
3020  
3025  
3030  
3035  
3040  
3045  
3050  
3055  
3060  
3065  
3070  
3075  
3080  
3085  
3090  
3095  
3100  
3105  
3110  
3115  
3120  
3125  
3130  
3135  
3140  
3145  
3150  
3155  
3160  
3165  
3170  
3175  
3180  
3185  
3190  
3195  
3200  
3205  
3210  
3215  
3220  
3225  
3230  
3235  
3240  
3245  
3250  
3255  
3260  
3265  
3270  
3275  
3280  
3285  
3290  
3295  
3300  
3305  
3310  
3315  
3320  
3325  
3330  
3335  
3340  
3345  
3350  
3355  
3360  
3365  
3370  
3375  
3380  
3385  
3390  
3395  
3400  
3405  
3410  
3415  
3420  
3425  
3430  
3435  
3440  
3445  
3450  
3455  
3460  
3465  
3470  
3475  
3480  
3485  
3490  
3495  
3500  
3505  
3510  
3515  
3520  
3525  
3530  
3535  
3540  
3545  
3550  
3555  
3560  
3565  
3570  
3575  
3580  
3585  
3590  
3595  
3600  
3605  
3610  
3615  
3620  
3625  
3630  
3635  
3640  
3645  
3650  
3655  
3660  
3665  
3670  
3675  
3680  
3685  
3690  
3695  
3700  
3705  
3710  
3715  
3720  
3725  
3730  
3735  
3740  
3745  
3750  
3755  
3760  
3765  
3770  
3775  
3780  
3785  
3790  
3795  
3800  
3805  
3810  
3815  
3820  
3825  
3830  
3835  
3840  
3845  
3850  
3855  
3860  
3865  
3870  
3875  
3880  
3885  
3890  
3895  
3900  
3905  
3910  
3915  
3920  
3925  
3930  
3935  
3940  
3945  
3950  
3955  
3960  
3965  
3970  
3975  
3980  
3985  
3990  
3995  
4000  
4005  
4010  
4015  
4020  
4025  
4030  
4035  
4040  
4045  
4050  
4055  
4060  
4065  
4070  
4075  
4080  
4085  
4090  
4095  
4100  
4105  
4110  
4115  
4120  
4125  
4130  
4135  
4140  
4145  
4150  
4155  
4160  
4165  
4170  
4175  
4180  
4185  
4190  
4195  
4200  
4205  
4210  
4215  
4220  
4225  
4230  
4235  
4240  
4245  
4250  
4255  
4260  
4265  
4270  
4275  
4280  
4285  
4290  
4295  
4300  
4305  
4310  
4315  
4320  
4325  
4330  
4335  
4340  
4345  
4350  
4355  
4360  
4365  
4370  
4375  
4380  
4385  
4390  
4395  
4400  
4405  
4410  
4415  
4420  
4425  
4430  
4435  
4440  
4445  
4450  
4455  
4460  
4465  
4470  
4475  
4480  
4485  
4490  
4495  
4500  
4505  
4510  
4515  
4520  
4525  
4530  
4535  
4540  
4545  
4550  
4555  
4560  
4565  
4570  
4575  
4580  
4585  
4590  
4595  
4600  
4605  
4610  
4615  
4620  
4625  
4630  
4635  
4640  
4645  
4650  
4655  
4660  
4665  
4670  
4675  
4680  
4685  
4690  
4695  
4700  
4705  
4710  
4715  
4720  
4725  
4730  
4735  
4740  
4745  
4750  
4755  
4760  
4765  
4770  
4775  
4780  
4785  
4790  
4795  
4800  
4805  
4810  
4815  
4820  
4825  
4830  
4835  
4840  
4845  
4850  
4855  
4860  
4865  
4870  
4875  
4880  
4885  
4890  
4895  
4900  
4905  
4910  
4915  
4920  
4925  
4930  
4935  
4940  
4945  
4950  
4955  
4960  
4965  
4970  
4975  
4980  
4985  
4990  
4995  
5000  
5005  
5010  
5015  
5020  
5025  
5030  
5035  
5040  
5045  
5050  
5055  
5060  
5065  
5070  
5075  
5080  
5085  
5090  
5095  
5100  
5105  
5110  
5115  
5120  
5125  
5130  
5135  
5140  
5145  
5150  
5155  
5160  
5165  
5170  
5175  
5180  
5185  
5190  
5195  
5200  
5205  
5210  
5215  
5220  
5225  
5230  
5235  
5240  
5245  
5250  
5255  
5260  
5265  
5270  
5275  
5280  
5285  
5290  
5295  
5300  
5305  
5310  
5315  
5320  
5325  
5330  
5335  
5340  
5345  
5350  
5355  
5360  
5365  
5370  
5375  
5380  
5385  
5390  
5395  
5400  
5405  
5410  
5415  
5420  
5425  
5430  
5435  
5440  
5445  
5450  
5455  
5460  
5465  
5470  
5475  
5480  
5485  
5490  
5495  
5500  
5505  
5510  
5515  
5520  
5525  
5530  
5535  
5540  
5545  
5550  
5555  
5560  
5565  
5570  
5575  
5580  
5585  
5590  
5595  
5600  
5605  
5610  
5615  
5620  
5625  
5630  
5635  
5640  
5645  
5650  
5655  
5660  
5665  
5670  
5675  
5680  
5685  
5690  
5695  
5700  
5705  
5710  
5715  
5720  
5725  
5730  
5735  
5740  
5745  
5750  
5755  
5760  
5765  
5770  
5775  
5780  
5785  
5790  
5795  
5800  
5805  
5810  
5815  
5820  
5825  
5830  
5835  
5840  
5845  
5850  
5855  
5860  
5865  
5870  
5875  
5880  
5885  
5890  
5895  
5900  
5905  
5910  
5915  
5920  
5925  
5930  
5935  
5940  
5945  
5950  
5955  
5960  
5965  
5970  
5975  
5980  
5985  
5990  
5995  
6000  
6005  
6010  
6015  
6020  
6025  
6030  
6035  
6040  
6045  
6050  
6055  
6060  
6065  
6070  
6075  
6080  
6085  
6090  
6095  
6100  
6105  
6110  
6115  
6120  
6125  
6130  
6135  
6140  
6145  
6150  
6155  
6160  
6165  
6170  
6175  
6180  
6185  
6190  
6195  
6200  
6205  
6210  
6215  
6220  
6225  
6230  
6235  
6240  
6245  
6250  
6255  
6260  
6265  
6270  
6275  
6280  
6285  
6290  
6295  
6300  
6305  
6310  
6315  
6320  
6325  
6330  
6335  
6340  
6345  
6350  
6355  
6360  
6365  
6370  
6375  
6380  
6385  
6390  
6395  
6400  
6405  
6410  
6415  
6420  
6425  
6430  
6435  
6440  
6445  
6450  
6455  
6460  
6465  
6470  
6475  
6480  
6485  
6490  
6495  
6500  
6505  
6510  
6515  
6520  
6525  
6530  
6535  
6540  
6545  
6550  
6555  
6560  
6565  
6570  
6575  
6580  
6585  
6590  
6595  
6600  
6605  
6610  
6615  
6620  
6625  
6630  
6635  
6640  
6645  
6650  
6655  
6660  
6665  
6670  
6675  
6680  
6685  
6690  
6695  
6700  
6705  
6710  
6715  
6720  
6725  
6730  
6735  
6740  
6745  
6750  
6755  
6760  
6765  
6770  
6775  
6780  
6785  
6790  
6795  
6800  
6805  
6810  
6815  
6820  
6825  
6830  
6835  
6840  
6845  
6850  
6855  
6860  
6865  
6870  
6875  
6880  
6885  
6890  
6895  
6900  
6905  
6910  
6915  
6920  
6925  
6930  
6935  
6940  
6945  
6950  
6955  
6960  
6965  
6970  
6975  
6980  
6985  
6990  
6995  
7000  
7005  
7010  
7015  
7020  
7025  
7030  
7035  
7040  
7045  
7050  
7055  
7060  
7065  
7070  
7075  
7080  
7085  
7090  
7095  
7100  
7105  
7110  
7115  
7120  
7125  
7130  
7135  
7140  
7145  
7150  
7155  
7160  
7165  
7170  
7175  
7180  
7185  
7190  
7195  
7200  
7205  
7210  
7215  
7220  
7225  
7230  
7235  
7240  
7245  
7250  
7255  
7260  
7265  
7270  
7275  
7280  
7285  
7290  
7295  
7300  
7305  
7310  
7315  
7320  
7325  
7330  
7335  
7340  
7345  
7350  
7355  
7360  
7365  
7370  
7375  
7380  
7385  
7390  
7395  
7400  
7405  
7410  
7415  
7420  
7425  
7430  
7435  
7440  
7445  
7450  
7455  
7460  
7465  
7470  
7475  
7480  
7485  
7490  
7495  
7500  
7505  
7510  
7515  
7520  
7525  
7530  
7535  
7540  
7545  
7550  
7555  
7560  
7565  
7570  
7575  
7580  
7585  
7590  
7595  
7600  
7605  
7610  
7615  
7620  
7625  
7630  
7635  
7640  
7645  
7650  
7655  
7660  
7665  
7670  
7675  
7680  
7685  
7690  
7695  
7700  
7705  
7710  
7715  
7720  
7725  
7730  
7735  
7740  
7745  
7750  
7755  
7760  
7765  
7770  
7775  
7780  
7785  
7790  
7795  
7800  
7805  
7810  
7815  
7820  
7825  
7830  
7835  
7840  
7845  
7850  
7855  
7860  
7865  
7870  
7875  
7880  
7885  
7890  
7895  
7900  
7905  
7910  
7915  
7920  
7925  
7930  
7935  
7940  
7945  
7950  
7955  
7960  
7965  
7970  
7975  
7980  
7985  
7990  
7995  
8000  
8005  
8010  
8015  
8020  
8025  
8030  
8035  
8040  
8045  
8050  
8055  
8060  
8065  
8070  
8075  
8080  
8085  
8090  
8095  
8100  
8105  
8110  
8115  
8120  
8125  
8130  
8135  
8140  
8145  
8150  
8155  
8160  
8165  
8170  
8175  
8180  
8185  
8190  
8195  
8200  
8205  
8210  
8215  
8220  
8225  
8230  
8235  
8240  
8245  
8250  
8255  
8260  
8265  
8270  
8275  
8280  
8285  
8290  
8295  
8300  
8305  
8310  
8315  
8320  
8325  
8330  
8335  
8340  
8345  
8350  
8355  
8360  
8365  
8370  
8375  
8380  
8385  
8390  
8395  
8400  
8405  
8410  
8415  
8420  
8425  
8430  
8435  
8440  
8445  
8450  
8455  
8460  
8465  
8470  
8475  
8480  
8485  
8490  
8495  
8500  
8505  
8510  
8515  
8520  
8525  
8530  
8535  
8540  
8545  
8550  
8555  
8560  
8565  
8570  
8575  
8580  
8585  
8590  
8595  
8600  
8605  
8610  
8615  
8620  
8625  
8630  
8635  
8640  
8645  
8650  
8655  
8660  
8665  
8670  
8675  
8680  
8685  
8690  
8695  
8700  
8705  
8710  
8715  
8720  
8725  
8730  
8735  
8740  
8745  
8750  
8755  
8760  
8765  
8770  
8775  
8780  
8785  
8790  
8795  
8800  
8805  
8810  
8815  
8820  
8825  
8830  
8835  
8840  
8845  
8850  
8855  
8860  
8865  
8870  
8875  
8880  
8885  
8890  
8895  
8900  
8905  
8910  
8915  
8920  
8925  
8930  
8935  
8940  
8945  
8950  
8955  
8960  
8965  
8970  
8975  
8980  
8985  
8990  
8995  
9000  
9005  
9010  
9015  
9020  
9025  
9030  
9035  
9040  
9045  
9050  
9055  
9060  
9065  
9070  
9075  
9080  
9085  
9090  
9095  
9100  
9105  
9110  
9115  
9120  
9125  
9130  
9135  
9140  
9145  
9150  
9155  
9160  
9165  
9170  
9175  
9180  
9185  
9190  
9195  
9200  
9205  
9210  
9215  
9220  
9225  
9230  
9235  
9240  
9245  
9250  
9255  
9260  
9265  
9270  
9275  
9280  
9285  
9290  
9295  
9300  
9305  
9310  
9315  
9320  
9325  
9330  
9335  
9340  
9345  
9350  
9355  
9360  
9365  
9370  
9375  
9380  
9385  
9390  
9395  
9400  
9405  
9410  
9415  
9420

source address translation table for the source (inbound) domain, in step 1026, based upon the source (inbound) domain for the packet determined in step 1022. The NAT 102 then searches the source address translation table for a source address translation table entry mapping the source host local address in the source (inbound) address domain to the source host global address for the destination (outbound) address domain, in step 1028, specifically by searching the source address translation table for a source address translation table entry having a Source Local Address field equal to the source host local address and a Destination Address Domain field equal to the destination (outbound) domain determined in step 1024.

If the source address translation table entry is found (YES in step 1030), then the NAT 102 proceeds to translate the source address in the packet from the source host local address into the source host global address for the destination (outbound) address domain, in step 1038. In particular, the NAT 102 obtains the source host global address from the Source Global Address field of the source address translation table entry, and replaces the source host local address in the packet with the source host global address. The source address translation logic then terminates in step 1040.

However, if the source address translation table entry is not found (NO in step 1030), then the NAT 102 dynamically allocates a source host global address for the destination address domain, creates the appropriate address translation entries, and translates the source address in the packet by replacing the source host local address in the packet with the dynamically allocated source host global address. In particular, the NAT 102 first selects a unique source host global address from a pool of network addresses, in step 1032. The NAT 102 then creates a source address translation table entry in the source (inbound) address domain's source address translation table mapping the source host local address in the source (inbound) address domain to the source host global address for the destination (outbound) address domain, in step 1034, and creates a corresponding destination address translation table entry in the destination address translation table mapping the source host global address to the source host local address in the source (inbound) address domain, in step 1036. The NAT 102 then translates the source address in the packet from the source host local address into the source host global address for the destination (outbound) address domain, in step 1038, specifically by replacing the source host local address in the packet with the source host

global address. The source address translation logic then terminates in step 1040.

FIG. 11A is a block diagram showing the relevant logic blocks of an exemplary NAT 102. The NAT 102 is operably coupled to at least a source (inbound) address domain of the communication network 100 by way of a Source (Inbound) Network Interface 1110 and to a destination (outbound) address domain of the communication network 100 by way of a Destination (Outbound) Network Interface 1150. Packets received over the Source (Inbound) Network Interface 1110 are processed by a Packet Processor 1130. The Packet Processor 1130 is operably coupled to perform any necessary address translations on the packet. The translated packets are forwarded to the destination (outbound) address domain via the Destination (Outbound) Network Interface 1150.

FIG. 11B shows the relevant logic blocks of an exemplary Packet Processor 1130. The Packet Processor 1130 includes both destination address translation logic (1136, 1137) and source address translation logic (1132, 1133). The destination address translation logic translates a destination host global address into a destination host local address in the destination (outbound) address domain, if such a translation is determined to be required. The source address translation logic translates a source host local address in the source (inbound) address domain into a source host global address for the destination (outbound) address domain, if such a translation is determined to be required. It should be noted that the destination address translation logic and the source address translation logic are shown as being operably coupled in parallel for convenience only. In a preferred embodiment of the present invention, the source address translation logic operates after completion of the destination address translation logic, and preferably obtains the source (inbound) address domain and the destination (outbound) address domain from the destination address translation table entry that is used by the destination address translation logic for translating the destination address in the packet.

The destination address translation logic determines whether the destination address requires translation, and translates the destination address from a destination host global address into a destination host local address if destination address translation is required. Specifically, the packet is processed by a Destination Address Filter 1136, which determines whether or not the destination address in the packet requires translation. The Destination

Address Filter 1136 may utilize address translation information stored in the Address Translation Table(s) 1134, and particularly in a destination address translation table, in order to determine whether or not the destination address in the packet requires translation. If the Destination Address Filter 1136 determines that the destination address in the packet does not require address translation, then the Destination Address Filter 1136 forwards the packet unchanged via the path 1138. However, assuming that the Destination Address Filter 1136 determines that the destination address in the packet does require address translation, then the Destination Address Translator 1137 translates the destination address from the destination host global address into the destination host local address in the destination (outbound) address domain, specifically by finding a destination address translation table entry in the Address Translation Table(s) 1134 corresponding to the destination host global address, obtaining the destination host local address from the destination address translation table entry, and inserting the destination host local address into the destination address field of the packet.

The source address translation logic determines whether the source address requires translation, and translates the source address from a source host local address into a source host global address for the destination (outbound) address domain if source address translation is required. Specifically, the packet is processed by a Source Address Filter 1132, which determines whether or not the source address in the packet requires translation. The Source Address Filter 1132 may utilize address translation information stored in the Address Translation Table(s) 1134 to determine whether or not the source address in the packet requires translation. If the Source Address Filter 1132 determines that the source address in the packet does not require address translation, then the Source Address Filter 1132 forwards the packet unchanged via the path 1131. However, assuming that the Source Address Filter 1132 determines that the source address in the packet does require address translation, then the Source Address Translator 1133 translates the source address from the source host local address into the source host global address for the destination (outbound) address domain, specifically by selecting a source address translation table for the source (inbound) address domain, searching the source address translation table for a source address translation table entry corresponding to the source host local address and the destination (outbound) address

domain, obtaining the source host global address from the source address translation table entry, and inserting the source host global address into the source address field of the packet.

In an exemplary embodiment of the invention, the Packet Processor 1130 is managed through a Management Information Base (MIB) 1170, which is accessible through a Management Agent 1160 such as a Simple Network Management Protocol (SNMP) agent. The MIB 1170 defines various management objects for configuring and controlling various network address translation functions. Specifically, an exemplary MIB 1170 defines management objects for configuring and controlling the set of interfaces that participate in network address translation, configuring and controlling ranges of addresses to be translated, configuring and controlling ranges of addresses for address pools, configuring and controlling static address mappings, and monitoring the current set of address translations that are in effect. It should be noted that the MIB 1170 is an abstraction for management information and that the management objects defined by the MIB 1170 do not necessarily correspond to any particular management information or management information format maintained within the NAT 102.

An exemplary MIB 1170 for configuring and controlling the Packet Processor 1130 is shown in FIGs. 12A-12P, and is included herein for convenience:

Wellfleet-NAT-MIB DEFINITIONS ::= BEGIN

20

--

IMPORTS

25       IpAddress, Counter, Gauge

          FROM RFC1155-SMI

          OBJECT-TYPE

          FROM RFC-1212

          DisplayString

30        FROM RFC1213-MIB

wfNatGroup  
FROM Wellfleet-COMMON-MIB;

wfNatIfTable OBJECT-TYPE

SYNTAX SEQUENCE OF WfNatIfEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"The set of interface that are participating  
in the NAT protocol."

::= { wfNatGroup 6 }

wfNatIfEntry OBJECT-TYPE

SYNTAX WfNatIfEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"An single instance of a NAT interface entry."

INDEX { wfNatIfIpAddress,  
wfNatIfCircuit }

::= { wfNatIfTable 1 }

WfNatIfEntry ::= SEQUENCE {

wfNatIfDelete

INTEGER,

wfNatIfDisable

INTEGER,

wfNatIfIpAddress

IpAddress,

wfNatIfCircuit

5                   INTEGER,  
wfNatIfType  
                  INTEGER,  
wfNatIfState  
                  INTEGER,  
wfNatIfTxCount  
                  Counter,  
wfNatIfRxCount  
                  Counter,  
wfNatIfPktDropCount  
                  Counter,  
wfNatIfDomain  
                  DisplayString  
                  }  
#5

20 wfNatIfDelete OBJECT-TYPE

SYNTAX        INTEGER {  
                  created(1),  
                  deleted(2)  
                  }

25 ACCESS        read-write  
STATUS         mandatory

DEFVAL         { created }  
                  := { wfNatIfEntry 1 }

30 wfNatIfDisable OBJECT-TYPE

SYNTAX        INTEGER {

```
    enabled(1),  
    disabled(2)  
}  
5      ACCESS      read-write  
      STATUS      mandatory
```

**DESCRIPTION**

"The NAT interface's administrative status. The value 'enabled' denotes that NAT has been configured on the interface. The value 'disabled' denotes that the interface is not running NAT."

```
DEFVAL      { enabled }  
::= { wfNatIfEntry 2 }
```

**wfNatIfIpAddress OBJECT-TYPE**

```
SYNTAX      IpAddress  
ACCESS      read-only  
STATUS      mandatory
```

**DESCRIPTION**

"The IP address of this NAT interface."

```
20      ::= { wfNatIfEntry 3 }
```

**wfNatIfCircuit OBJECT-TYPE**

```
SYNTAX      INTEGER  
ACCESS      read-write  
25      STATUS      mandatory
```

**DESCRIPTION**

"The circuit number of this interface."

```
 ::= { wfNatIfEntry 4 }
```

**30 wfNatIfType OBJECT-TYPE**

5                   SYNTAX    INTEGER {  
                          uniDirInbound(1),  
                          uniDirOutbound(2),  
                          biDirectional(3)  
                          }  
                         ACCESS    read-write  
                         STATUS    mandatory  
                         DESCRIPTION  
                          "Denotes the type of NAT interface being defined."  
                         DEFVAL{ uniDirInbound }  
                         ::= { wfNatIfEntry 5 }

10                  wfNatIfState OBJECT-TYPE  
                         SYNTAX    INTEGER {  
                          up(1),  
                          down(2),  
                          init(3)  
                          }  
                         ACCESS    read-only  
                         STATUS    mandatory  
                         DESCRIPTION  
                          "The state of NAT on this interface"  
                         DEFVAL    { down }  
                         ::= { wfNatIfEntry 6 }

15  
20  
25  
30                  wfNatIfTxCount OBJECT-TYPE  
                         SYNTAX    Counter  
                         ACCESS    read-only  
                         STATUS    mandatory  
                         DESCRIPTION

"Number of packets mapped over this interface from  
the local to the global network."

::= { wfNatIfEntry 7 }

5

wfNatIfRxCount OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"Number of packets mapped over this interface from  
the global to the local network."

::= { wfNatIfEntry 8 }

10  
11  
12  
13  
14  
15

wfNatIfPktDropCount OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"Number of packets dropped on this interface"

20

::= { wfNatIfEntry 9 }

wfNatIfDomain OBJECT-TYPE

SYNTAX DisplayString

ACCESS read-write

25

STATUS mandatory

DESCRIPTION

"When wfNatIfType is set to biDirectional, specifies  
the Address Domain Name that this interface is  
connected to, otherwise set to null."

30

::= { wfNatIfEntry 10 }

5           **wfNatAddressRangeTable** OBJECT-TYPE

SYNTAX SEQUENCE OF WfNatAddressRangeEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"Table of address ranges. "

::= { wfNatGroup 8 }

10           **wfNatAddressRangeEntry** OBJECT-TYPE

SYNTAX WfNatAddressRangeEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"Information describing each of the available address ranges."

INDEX { wfNatAddressRangeAddress,  
          wfNatAddressRangePrefixLen,  
          wfNatAddressRangeIndex }  
 ::= { wfNatAddressRangeTable 1 }

20           **WfNatAddressRangeEntry** ::= SEQUENCE {

**wfNatAddressRangeDelete**

          INTEGER,

**wfNatAddressRangeDisable**

          INTEGER,

**wfNatAddressRangeAddress**

          IpAddress,

**wfNatAddressRangePrefixLen**

          INTEGER,

**wfNatAddressRangeIndex**

5                   INTEGER,  
wfNatAddressRangeNto1Addr  
                  IpAddress,  
wfNatAddressRangeType  
                  INTEGER,  
wfNatAddressRangeDomain  
                  DisplayString,  
wfNatAddressRangeTransPool  
                  INTEGER,  
wfNatAddressRangeStaticNextHop  
                  IpAddress,  
wfNatAddressRangeUnnumCct  
                  INTEGER  
                  }  
E5

wfNatAddressRangeDelete OBJECT-TYPE

20                   SYNTAX INTEGER {  
                  created(1),  
                  deleted(2)  
                  }

                  ACCESS read-write

                  STATUS mandatory

                  DESCRIPTION

25                   "Create/Delete parameter. Default is created. Users perform  
                  a set operation on this object in order to create/delete  
                  an address range entry."

                  DEFVAL { created }  
                  ::= { wfNatAddressRangeEntry 1 }

```
wfNatAddressRangeDisable OBJECT-TYPE
  SYNTAX  INTEGER {
    enabled(1),
    disabled(2)
  }
  ACCESS  read-write
  STATUS   mandatory
  DESCRIPTION
    "Enable/Disable parameter. Default is enabled. Users perform
     a set operation on this object in order to enable/disable
     an address range entry."
  DEFVAL { enabled }
  ::= { wfNatAddressRangeEntry 2 }
```

SYNTAX ipAddress

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The IP beginning address of this range

**::= { wfNatAddressRangeEntry 3 }**

wfNatAddressRangePrefixLen OBJECT-TYPE  
    SYNTAX INTEGER (1 .. 32)  
    ACCESS read-only  
    STATUS mandatory  
    DESCRIPTION  
        "The number of contiguous bits set in the IP address mask  
            which are used to define the address range of the entry."  
     ::= { wfNatAddressRangeEntry 4 }

5           wfNatAddressRangeIndex OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

DESCRIPTION

"A unique value for this entry in wfNatAddressRangeTable."

::= { wfNatAddressRangeEntry 5 }

10           wfNatAddressRangeNto1Addr OBJECT-TYPE

SYNTAX IpAddress

ACCESS read-write

STATUS mandatory

DESCRIPTION

"When wfNatAddressRangeType is set to srcAddrFilter,  
specifies the N-to-1 translation address used for  
this range, otherwise set to zero."

DEFVAL { 0 }

::= { wfNatAddressRangeEntry 6 }

20           wfNatAddressRangeType OBJECT-TYPE

SYNTAX INTEGER {

    sourceAddrFilter(1),  
    translationPool(2),  
    domainSrcAddrFilter(3),  
    domainTransPool(4)

}

ACCESS read-write

STATUS mandatory

DESCRIPTION

"Denotes the type of address range being defined."

sourceAddrFilter: a range of IP addresses used to detect packets  
which need traditional NAT forwarding.

translationPool: for traditional NAT forwarding, a range of  
IP addresses from which translation addresses are picked.

5 domainSrcAddrFilter: a range of IP addresses used to detect domain  
specific packets which need domain specific NAT forwarding.

domainTransPool: for domain specific NAT forwarding, a range of  
IP addresses from which domain specific translation addresses  
are picked."

10 DEFVAL{ sourceAddrFilter }  
::= { wfNatAddressRangeEntry 7 }

15 wfNatAddressRangeDomain OBJECT-TYPE

SYNTAX DisplayString

ACCESS read-write

STATUS mandatory

DESCRIPTION

20 "When wfNatAddressRangeType is set to domainSrcAddrFilter or  
domainTransPool, specifies the Address Domain Name that this  
address range is valid for, otherwise set to null."

25 ::= { wfNatAddressRangeEntry 8 }

wfNatAddressRangeTransPool OBJECT-TYPE

SYNTAX INTEGER {

25 inbound(1),

outbound(2)

}

ACCESS read-write

STATUS mandatory

30 DESCRIPTION

"This attribute is only valid for the wfNatAddressRangeType as domainSrcAddrFilter(3). The value of this attribute decides where to get the translation address for this range from.

5 This could be either the translation Pool defined for the inbound domain or the outbound domain for the packet in question."

DEFVAL{ outbound }  
::= { wfNatAddressRangeEntry 9 }

10 wfNatAddressRangeStaticNextHop OBJECT-TYPE

SYNTAX IpAddress  
ACCESS read-write  
STATUS mandatory  
DESCRIPTION

15 "The IP address of the next hop of this range entry."

DEFVAL{ 0 }  
::= { wfNatAddressRangeEntry 10 }

20 wfNatAddressRangeUnnumCct OBJECT-TYPE

SYNTAX INTEGER  
ACCESS read-write  
STATUS mandatory  
DESCRIPTION

25 "This Nat Address range over the unnumbered interface."

DEFVAL{ 0 }  
::= { wfNatAddressRangeEntry 11 }

30 wfNatStaticMappingTable OBJECT-TYPE

SYNTAX SEQUENCE OF WfNatStaticMappingEntry  
ACCESS not-accessible

5 STATUS mandatory

10 DESCRIPTION

15 "This table creates instances of pre-defined NAT translations."

20 ::= { wfNatGroup 9 }

25

30 wfNatStaticMappingEntry OBJECT-TYPE

35 SYNTAX WfNatStaticMappingEntry

40 ACCESS not-accessible

45 STATUS mandatory

50 DESCRIPTION

55 "A single original source address to translated address  
60 translation."

65 INDEX { wfNatStaticMappingTransAddress,

70 wfNatStaticMappingProtocol,

75 wfNatStaticMappingTransPort }

80 ::= { wfNatStaticMappingTable 1 }

85 WfNatStaticMappingEntry ::= SEQUENCE {

90 wfNatStaticMappingDelete

95 INTEGER,

100 wfNatStaticMappingDisable

105 INTEGER,

110 wfNatStaticMappingOrigAddress

115 IpAddress,

120 wfNatStaticMappingTransAddress

125 IpAddress,

130 wfNatStaticMappingProtocol

135 INTEGER,

140 wfNatStaticMappingOrigPort

145 INTEGER,

150

5           wfNatStaticMappingTransPort  
              INTEGER,  
              wfNatStaticMappingInDomain  
              DisplayString,  
              wfNatStaticMappingOutDomain  
              DisplayString,  
              wfNatStaticMappingStaticNextHop  
              IpAddress,  
              wfNatStaticMappingUnnumCct  
              INTEGER  
              }  
10

15           wfNatStaticMappingDelete OBJECT-TYPE  
              SYNTAX INTEGER {  
                  created(1),  
                  deleted(2)  
              }  
              ACCESS read-write  
              STATUS mandatory  
20           DESCRIPTION  
              "Create/Delete parameter. Default is created. Users perform a set  
              operation on this object in order to create/delete a static  
              address translation entry."  
              DEFVAL { created }  
25           ::= { wfNatStaticMappingEntry 1 }

30           wfNatStaticMappingDisable OBJECT-TYPE  
              SYNTAX INTEGER {  
                  enabled(1),  
                  disabled(2)

}

ACCESS read-write

STATUS mandatory

DESCRIPTION

5 "Enable/Disable parameter. Default is enabled. Users perform a  
set operation on this object in order to enable/disable a static  
address translation entry."

DEFVAL { enabled }

::= { wfNatStaticMappingEntry 2 }

10 wfNatStaticMappingOrigAddress OBJECT-TYPE

SYNTAX InetAddress

ACCESS read-write

STATUS mandatory

15 DESCRIPTION

"The original (un-translated) address of the translation."

::= { wfNatStaticMappingEntry 3 }

20 wfNatStaticMappingTransAddress OBJECT-TYPE

SYNTAX InetAddress

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The translated address of the translation."

25 ::= { wfNatStaticMappingEntry 4 }

30 wfNatStaticMappingProtocol OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

5  
DESCRIPTION

"The IP protocol of the translation. Example values are  
6 for TCP, and 17 for UDP."

10  
::= { wfNatStaticMappingEntry 5 }

15  
wfNatStaticMappingOrigPort OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-write

STATUS mandatory

20  
DESCRIPTION

"The original (domain specific) UDP or TCP port of the  
translation. This will only be relevant if the protocol  
is either UDP or TCP."

15  
::= { wfNatStaticMappingEntry 6 }

20  
wfNatStaticMappingTransPort OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

25  
DESCRIPTION

"The translated UDP or TCP port of the translation. This will  
only be relevant if the protocol is either UDP or TCP."

15  
::= { wfNatStaticMappingEntry 7 }

25  
wfNatStaticMappingInDomain OBJECT-TYPE

SYNTAX DisplayString

ACCESS read-write

STATUS mandatory

30  
DESCRIPTION

"This attribute specifies the name of the address domain that this

source translation shall be valid for. In other words, this translation shall only be valid for source addresses coming inbound from this domain."

DEFVAL { "private" }

5 ::= { wfNatStaticMappingEntry 8 }

wfNatStaticMappingOutDomain OBJECT-TYPE

SYNTAX DisplayString

ACCESS read-write

10 STATUS mandatory

DESCRIPTION

"This attribute specifies the name of the outbound address domain that this translation will be valid for. In other words, this translation only applies to translations that will be forwarded out into this address domain."

DEFVAL { "public" }

15 ::= { wfNatStaticMappingEntry 9 }

wfNatStaticMappingStaticNextHop OBJECT-TYPE

20 SYNTAX IpAddress

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The IP address of the next hop of this static entry."

25 DEFVAL{ 0 }

::= { wfNatStaticMappingEntry 10 }

wfNatStaticMappingUnnumCct OBJECT-TYPE

SYNTAX INTEGER

30 ACCESS read-write

5 STATUS mandatory

DESCRIPTION

"This Nat static translation over the unnumbered interface."

DEFVAL{ 0 }

5 ::= { wfNatStaticMappingEntry 11 }

10 wfNatMappingTable OBJECT-TYPE

SYNTAX SEQUENCE OF WfNatMappingEntry

ACCESS not-accessible

15 STATUS mandatory

DESCRIPTION

"This table defines the current set of address translations  
that are in effect."

20 ::= { wfNatGroup 10 }

25 wfNatMappingEntry OBJECT-TYPE

SYNTAX WfNatMappingEntry

ACCESS not-accessible

STATUS mandatory

20 DESCRIPTION

"A single original source address to translated address  
translation."

25 INDEX { wfNatMappingTransAddress,

wfNatMappingProtocol,

wfNatMappingTransPort }

25 ::= { wfNatMappingTable 1 }

30 WfNatMappingEntry ::= SEQUENCE {

wfNatMappingOrigAddress

IpAddress,

5           wfNatMappingTransAddress  
              IpAddress,  
              wfNatMappingProtocol  
              INTEGER,  
              wfNatMappingOrigPort  
              INTEGER,  
              wfNatMappingTransPort  
              INTEGER,  
              wfNatMappingTxCount  
              Counter,  
              wfNatMappingRxCount  
              Counter,  
              wfNatMappingTimeout  
              Counter,  
              wfNatMappingMode  
              INTEGER,  
              wfNatMappingInDomain  
              DisplayString,  
              wfNatMappingOutDomain  
20           DisplayString  
              }

25           wfNatMappingOrigAddress OBJECT-TYPE  
              SYNTAX  IpAddress  
              ACCESS  read-only  
              STATUS  mandatory  
              DESCRIPTION  
              "The original (un-translated) address of the translation."  
              ::= { wfNatMappingEntry 1 }

5           wfNatMappingTransAddress OBJECT-TYPE

SYNTAX ipAddress

ACCESS read-only

STATUS mandatory

10           DESCRIPTION

      "The translated address of the translation."

15           ::= { wfNatMappingEntry 2 }

20           wfNatMappingProtocol OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

25           DESCRIPTION

      "The IP protocol of the translation."

30           ::= { wfNatMappingEntry 3 }

20           wfNatMappingOrigPort OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

25           DESCRIPTION

      "When the translation is for a TCP connection, this denotes  
      the original TCP port number."

30           ::= { wfNatMappingEntry 4 }

25           wfNatMappingTransPort OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

30           DESCRIPTION

"When the translation is for a TCP connection, this denotes  
the translated TCP port number."

::= { wfNatMappingEntry 5 }

5 wfNatMappingTxCount OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"Number of packets forwarded by NAT using this translation."

::= { wfNatMappingEntry 6 }

10 wfNatMappingRxCount OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"Number of packets received by NAT using this translation."

::= { wfNatMappingEntry 7 }

20

wfNatMappingTimeout OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

25

DESCRIPTION

"The time in seconds since this translation entry was last used.

This is used to age out translation entries."

::= { wfNatMappingEntry 8 }

30

wfNatMappingMode OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

DESCRIPTION

5       "This is the bit mask representing the type of this translation.

          Each bit specifies the type as follows:

10       The translation could be only ONE of the following three...

15       0x01000000 - This translation is originated on this router,  
          i.e. this NAT router performed the translation.

20       0x02000000 - This translation is learned from the peer,  
          i.e. this translation was learned from the  
          peer using NAT Synchronization feature.

25       0x04000000 - This translation is owned,  
          i.e. it was originally learned from peer, but  
          this router received traffic which used this  
          translation.

30       ...and only ONE of the following three.

          0x00000010 - This translation is the STATIC translation.

          0x00000020 - This translation is Dynamic(1 to 1) translation.

          0x00000040 - This translation is N to 1 translation."

25       DEFVAL { 0 }

           ::= { wfNatMappingEntry 9 }

30       wfNatMappingInDomain OBJECT-TYPE

          SYNTAX DisplayString

          ACCESS read-only

STATUS mandatory

## DESCRIPTION

"This attribute specifies the name of the address domain that this source translation shall be valid for. In other words, this translation shall only be valid for source addresses coming inbound from this domain."

::= { wfNatMappingEntry 10 }

## wfNatMappingOutDomain OBJECT-TYPE

## SYNTAX DisplayString

ACCESS read-only

STATUS mandatory

## DESCRIPTION

"This attribute specifies the name of the outbound address domain that this translation will be valid for. In other words, this translation only applies to translations that will be forwarded out into this address domain."

::= { wfNatMappingEntry 11 }

END -- Wellfleet-NPT-MIB

The MIB 1170 defines various management objects that are organized as tables within the MIB 1170. Specifically, the MIB 1170 defines a NAT interface table (wfNatIfTable), a NAT address range table (wfNatAddressRangeTable), a NAT static mapping table (wfNatStaticMappingTable), and a NAT mapping table (wfNatMappingTable).

The NAT interface table (wfNatIfTable) is used to configure and control the set of interfaces that are participating in network address translation. Each NAT interface table entry corresponds to a particular NAT interface, and includes a management object (wfNatIfDelete) to create or delete the table entry, a management object (wfNatIfDisable) to

enable or disable the table entry, a management object (wfNatIfIpAddress) indicating an IP address for the interface, a management object (wfNatIfCircuit) indicating a circuit number for the interface, a management object (wfNatIfType) indicating the interface type (unidirectional inbound, unidirectional outbound, bi-directional), a management object (wfNatIfState) indicating the interface state (up, down, init), a management object (wfNatIfTxCount) indicating the number of packets mapped over the interface from the local to the global network, a management object (wfNatIfRxCount) indicating the number of packets mapped over the interface from the global to the local network, a management object (wfNatIfPktDropCount) indicating the number of packets dropped on the interface, and a management object (wfNatIfDomain) indicating the address domain for the interface if the interface is configured as a bi-directional interface (i.e., wfNatIfType indicates bi-directional).

The NAT address range table (wfNatAddressRangeTable) is used to configure and control various address ranges. Different types of address ranges can be defined using the NAT address range table. One type of address range, referred to as a source address filter, defines a range of IP addresses that is used to detect packets that require traditional NAT forwarding. Another type of address range, referred to as a translation pool, defines a range of IP addresses from which translation addresses are selected for traditional NAT forwarding. Still another type of address range, referred to as a domain-specific source address filter, defines a range of IP addresses that is used to detect packets that require domain-specific NAT forwarding. Yet another type of address range, referred to as a domain-specific translation pool, defines a range of IP addresses from which translation addresses are selected for domain-specific NAT forwarding. It should be noted that, although the MIB 1170 uses a single table structure to define multiple types of address ranges, the NAT address range table does not necessarily map to any one or more tables stored within the NAT 102.

Each address range is defined by a beginning IP address and a prefix length indicating the number of contiguous bits set in the IP address mask that is used to define the address range. Each source address filter range includes, among other things, an N-to-1 translation address. Each domain-specific range (i.e., domain-specific source address filter and domain-specific translation pool) includes, among other things, a domain name indicating the domain associated with the address range. Each domain-specific source address filter range

additionally includes, among other things, a translation pool indicator indicating a domain-specific translation pool to be used for domain-specific address translations, and specifically indicating whether to use a domain-specific translation pool associated with the inbound or the outbound domain.

5 Thus, each NAT address range table entry defines an address range, and includes a management object (wfNatAddressRangeDelete) to create or delete the table entry, a management object (wfNatAddressRangeDisable) to enable or disable the table entry, a management object (wfNatAddressRangeAddress) indicating the beginning IP address for the address range, a management object (wfNatAddressRangePrefixLen) indicating the prefix length (i.e., the number of contiguous bits set in the IP address mask that are used to define the address range), a management object (wfNatAddressRangeIndex) indicating a unique value (index) for the table entry, a management object (wfNatAddressRangeNto1Addr) indicating an N-to-1 translation address if the address range type (i.e., wfNatAddressRangeType, described below) is source address filter, a management object (wfNatAddressRangeType) indicating the address range type (source address filter, translation pool, domain-specific source address filter, domain-specific translation pool, a management object (wfNatAddressRangeDomain) indicating an address domain for the address range if the address range type (i.e., wfNatAddressRangeType) is source address filter or translation pool, a management object (wfNatAddressRangeTransPool) indicates whether 10 to obtain translation addresses from a translation pool defined for the inbound domain or from a translation pool defined for the outbound domain if the address range type (i.e., wfNatAddressRangeType) is domain-specific source address filter, a management object (wfNatAddressRangeStaticNextHop) indicating the IP address of the next hop for the address range, and a management object (wfNatAddressRangeUnnumCct) indicating whether the 15 address range is associated with an unnumbered interface.

20

25

The NAT static mapping table (wfNatStaticMappingTable) is used to configure and control static translation mappings. Each NAT static mapping table entry represents a single static translation mapping, and includes a management object (wfNatStaticMappingDelete) to create or delete the table entry, a management object (wfNatStaticMappingDisable) to enable or disable the table entry, a management object (wfNatStaticMappingOrigAddress) indicating 30

the original (un-translated) address for the translation, a management object  
(wfNatStaticMappingTransAddress) indicating the translated address for the translation, a  
management object (wfNatStaticMappingProtocol) indicating the IP protocol (e.g., TCP,  
5 UDP) for the translation, a management object (wfNatStaticMappingOrigPort) indicating the  
original (UDP or TCP) port for the translation, a management object  
(wfNatStaticMappingTransPort) indicating the translated (UDP or TCP) port for the  
translation, a management object (wfNatStaticMappingInDomain) indicating the inbound  
10 domain for the translation, a management object (wfNatStaticMappingOutDomain) indicating  
the outbound domain for the translation, a management object  
(wfNatStaticMappingStaticNextHop) indicating the IP address of the next hop for the  
translation entry, and a management object (wfNatStaticMappingUnnumCct) indicating  
15 whether the address range is associated with an unnumbered interface.

The NAT mapping table (wfNatMappingTable) is used to monitor the current set of  
address translations that are in effect. Each NAT mapping table entry represents a particular  
address translation, and includes a management object (wfNatMappingOrigAddress) the  
original (un-translated) address of the translation, a management object  
(wfNatMappingTransAddress) indicating the translated address of the translation, a  
management object (wfNatMappingProtocol) indicating the IP protocol of the translation, a  
management object (wfNatMappingOrigPort) indicating the original (UDP or TCP) port for  
20 the translation, a management object (wfNatMappingTransPort) indicating the translated  
(UDP or TCP) port for the translation), a management object (wfNatMappingTxCount)  
indicating the number of packets forwarded by the NAT using the translation, a management  
object (wfNatMappingRxCount) indicating the number of packets received by the NAT using  
the translation, a management object (wfNatMappingTimeout) indicating the amount of time  
25 (in seconds) since the translation was last used, a management object (wfNatMappingMode)  
indicating the source of the translation (locally originated and owned, remotely originated and  
owned, remotely originated and locally owned) and the type of translation (static, dynamic 1-  
to-1, N-to-1), a management object (wfNatMappingInDomain) indicating the inbound domain  
for the translation, and a management object (wfNatMappingOutDomain) indicating the  
30 outbound domain for the translation.

Management information that is configured using the MIB 1170 is used to set up address translation entries in the address translation table(s) 1134, which are then used within the Packet Processor 1130 for performing domain-specific network address translation.

While the exemplary MIB 1170 uses a beginning address and prefix length to define an address range, the present invention is in no way limited to using a beginning address and prefix length to define an address range. There are many other ways to define an address range. For one example, an address range can be defined using a beginning address and an ending address for the address range, in which case a MIB would include a management object for defining the beginning address and a management object for defining the ending address. For another example, an address range can be defined using a beginning address and an address range size indicating the number of addresses in the address range, in which case a MIB would include a management object for defining the beginning address and a management object for defining the address range size. Other ways to define an address range using MIB management objects will become apparent to a skilled artisan. All such ways to define an address range using MIB management objects fall within the scope of the present invention as claimed below.

Furthermore, an address range is not limited to a number of contiguous addresses. An address range may include one or more non-contiguous addresses, in which case a MIB would include management objects for defining the individual addresses and/or groups of addresses in an address range. For example, a MIB may include management objects for entering individual addresses in the address range.

It should be noted that the term "router" is used herein to describe a communication device that may be used in a communication system, and should not be construed to limit the present invention to any particular communication device type. Thus, a communication device may include, without limitation, a bridge, router, bridge-router (brouter), switch, node, or other communication device.

It should also be noted that the term "packet" is used herein to describe a communication message that may be used by a communication device (*e.g.*, created, transmitted, received, stored, or processed by the communication device) or conveyed by a communication medium, and should not be construed to limit the present invention to any

particular communication message type, communication message format, or communication protocol. Thus, a communication message may include, without limitation, a frame, packet, datagram, user datagram, cell, or other type of communication message.

It should also be noted that the logic flow diagrams are used herein to demonstrate various aspects of the invention, and should not be construed to limit the present invention to any particular logic flow or logic implementation. The described logic may be partitioned into different logic blocks (e.g., programs, modules, functions, or subroutines) without changing the overall results or otherwise departing from the true scope of the invention. Often times, logic elements may be added, modified, omitted, performed in a different order, or implemented using different logic constructs (e.g., logic gates, looping primitives, conditional logic, and other logic constructs) without changing the overall results or otherwise departing from the true scope of the invention.

The present invention may be embodied in many different forms, including, but in no way limited to, computer program logic for use with a processor (e.g., a microprocessor, microcontroller, digital signal processor, or general purpose computer), programmable logic for use with a programmable logic device (e.g., a Field Programmable Gate Array (FPGA) or other PLD), discrete components, integrated circuitry (e.g., an Application Specific Integrated Circuit (ASIC)), or any other means including any combination thereof. In a typical embodiment of the present invention, predominantly all of the management agent logic 1160 and MIB 1170 is implemented as a set of computer program instructions that is converted into a computer executable form, stored as such in a computer readable medium, and executed by a microprocessor within the NAT 102 under the control of an operating system.

Computer program logic implementing all or part of the functionality previously described herein may be embodied in various forms, including, but in no way limited to, a source code form, a computer executable form, and various intermediate forms (e.g., forms generated by an assembler, compiler, linker, or locator). Source code may include a series of computer program instructions implemented in any of various programming languages (e.g., an object code, an assembly language, or a high-level language such as Fortran, C, C++, JAVA, or HTML) for use with various operating systems or operating environments. The source code may define and use various data structures and communication messages. The

source code may be in a computer executable form (e.g., via an interpreter), or the source code may be converted (e.g., via a translator, assembler, or compiler) into a computer executable form.

The computer program may be fixed in any form (e.g., source code form, computer executable form, or an intermediate form) either permanently or transitorily in a tangible storage medium, such as a semiconductor memory device (e.g., a RAM, ROM, PROM, EEPROM, or Flash-Programmable RAM), a magnetic memory device (e.g., a diskette or fixed disk), an optical memory device (e.g., a CD-ROM), or other memory device. The computer program may be fixed in any form in a signal that is transmittable to a computer using any of various communication technologies, including, but in no way limited to, analog technologies, digital technologies, optical technologies, wireless technologies, networking technologies, and internetworking technologies. The computer program may be distributed in any form as a removable storage medium with accompanying printed or electronic documentation (e.g., shrink wrapped software), preloaded with a computer system (e.g., on system ROM or fixed disk), or distributed from a server or electronic bulletin board over the communication system (e.g., the Internet or World Wide Web).

Hardware logic (including programmable logic for use with a programmable logic device) implementing all or part of the functionality previously described herein may be designed using traditional manual methods, or may be designed, captured, simulated, or documented electronically using various tools, such as Computer Aided Design (CAD), a hardware description language (e.g., VHDL or AHDL), or a PLD programming language (e.g., PALASM, ABEL, or CUPL).

Programmable logic may be fixed either permanently or transitorily in a tangible storage medium, such as a semiconductor memory device (e.g., a RAM, ROM, PROM, EEPROM, or Flash-Programmable RAM), a magnetic memory device (e.g., a diskette or fixed disk), an optical memory device (e.g., a CD-ROM), or other memory device. The programmable logic may be fixed in a signal that is transmittable to a computer using any of various communication technologies, including, but in no way limited to, analog technologies, digital technologies, optical technologies, wireless technologies, networking technologies, and internetworking technologies. The programmable logic may be distributed

as a removable storage medium with accompanying printed or electronic documentation (e.g., shrink wrapped software), preloaded with a computer system (e.g., on system ROM or fixed disk), or distributed from a server or electronic bulletin board over the communication system (e.g., the Internet or World Wide Web).

5 The present invention may be embodied in other specific forms without departing  
from the true scope of the invention. The described embodiments are to be considered in all  
respects only as illustrative and not restrictive.